## Remarks/Arguments

Applicant respectfully disagrees with Examiner's characterization of the Li reference as teaching "means for converting the second graphics object into still picture data if said overlap cue indicating said overlap between the first and the second graphics object is generated." (Office Action, page 2) However, Applicant has amended the claims to further clarify the distinction between the present claims and the cited art.

**35 U.S.C. §103**

Claims 1, 3-7, and 10, stand rejected under 35 U.S.C. §103(a) as being unpatentable over Valmiki et al. (U.S. Patent No. 6,636,222; hereinafter referred to as "Valmiki"), in view of Terao et al. (U.S. Patent Publication No. 2001/0055011; hereinafter referred to as "Terao"), in view of Li et al. (U.S. Patent Publication No. 2003/0043172, hereinafter referred to as "Li"), and in further view of Huang et al. (U.S. Patent Publication No. 2003/0169372; hereinafter referred to as "Huang").

It is respectfully asserted that none of Valmiki, Terao, Li, or Huang, alone or in combination, disclose:

> "mixing means for mixing said first digital stream, generated by said OSD processor and representing said first graphics object, and said second digital stream, generated from said still picture data converted from said second graphics object from said graphics memory responsive to said overlap cue indicating overlap between the first and second graphics objects, into a video signal,"

as described claim 1. Furthermore, the combination of Valmiki, Terao, Li, and Huang would not permit the display of two graphics objects under the described OSD limitations, as would the teaching of claim 1.

Among the problems addressed by the present invention are the limitations of certain OSD processors, in particular the limitation that certain OSD processors cannot display two graphic objects that overlap, or in some cases, share the same lines of the screen. To address this problem, the present application discloses an apparatus that

converts the second of two overlapped graphics into a stationary picture memory separate from the OSD path, which in turn is combined by a mixer with the OSD plane containing the first graphic. The end result is the display of the two overlapping graphics, despite the inability of the OSD processor to perform such display on its own.

In contrast, Valmiki is not concerned with working around the limitations of OSD processors, but instead with rendering compressed video data within an allotted number of clock cycles. More specifically, Valmiki teaches a system where "a video and graphics system processes video data including both analog video, e.g., NTSC/PAL/SECAM/S-video, and digital video, e.g., MPEG-2 video in SDTV or HDTV format. The video and graphics system includes a video decoder, which is capable of concurrently decoding multiple SLICEs of MPEG-2 video data. The video decoder includes multiple row decoding engines for decoding the MPEG-2 video data. Each row decoding engine concurrently decodes two or more rows of the MPEG-2 video data. The row decoding engines have a pipelined architecture for concurrently decoding multiple rows of MPEG-2 video data. The video decoder may be integrated on an integrated circuit chip with other video and graphics system components such as transport processors for receiving one or more compressed data streams and for extracting video data, and a video compositor for blending processed video data with graphics." (Valmiki Abstract)

Valmiki discloses that its "video decoder includes multiple row decoding engines," but does not describe detection of overlaps or movement of an object from an OSD plane to a picture plane in response. As such, Valmiki does not disclose means for converting a second graphics object into still picture data if an overlap is detected between first and second graphics objects, or mixing the resulting stream with a stream from an OSD. Thus, it is respectfully submitted that Valmiki fails to disclose "mixing means for mixing said first digital stream, generated by said OSD processor and representing said first graphics object, and said second digital stream, generated from said still picture data converted from said second graphics object from said graphics memory responsive to said overlap cue indicating overlap between the first and second graphics objects, into a video signal," as described in claim 1, and would fail to address the problem of an OSD processor which is unable to manage two graphic objects that overlap.

Terao teaches an invention whose goal is "to provide an information processor which, in case a component in which a moving picture is displayed is overlapped by another window, can apply display effect only to a region on which the moving picture is actually displayed." (Terao Abstract) Terao deals with improving an image displayed on a CRT display (Terao [0002] to [0004]), but applies that improvement only to the region of the moving picture that is displayed. (Terao [0007] and [0008])

The method of Terao comprises the steps of detecting overlapping windows, identifying the region of the moving picture that is not overlapped, and applying the image improvement to that region only. Terao does disclose means for detecting an overlap between a first graphics object (the moving picture) and a second graphics object (Terao, [0056] to [0059]), but does not disclose means for converting one of the graphics objects into still picture data if the overlap cue is generated. Instead, Terao indicates that the image improvement process is applied only to the visible region of the moving picture. (Terao, [0069] to [0091]) That moving picture is not converted into still picture data. In Terao, upon detection of an overlap, picture effects are applied to the visible region of the moving picture. The moving picture is not converted into a still picture. Therefore, Terao, like Valmiki, fails to disclose "mixing means for mixing said first digital stream, generated by said OSD processor and representing said first graphics object, and said second digital stream, generated from said still picture data converted from said second graphics object from said graphics memory responsive to said overlap cue indicating overlap between the first and second graphics objects, into a video signal," as described in claim 1, and would fail to address the problem of an OSD processor which is unable to manage two graphic objects that overlap.

Li describes a "method for extracting textual and graphical overlays from video sequences involves steps of detecting a potential overlay in a video sequence and then verifying that the potential overlay is an actual overlay. Detection of textual overlays involves wavelet decomposition and neural network processing, while detection of graphical overlays involves template matching. Verification of textual and graphical overlays involves spatial and/or temporal verification." (Li Abstract)

The Examiner asserts that "Li teaches of detecting overlays which can be segmented from the rest of the video and compressing the overlay as a static image resulting in a more readable overlay (See [0004]). Li further discloses that such extraction of the overlay from the video is useful to enable rapid retrieval of video segments and for applications such as compression, indexing, logo detection, and video manipulation to re-create video with modified content and makes it possible to modify the overlay independently from the underlying video without the need for time-consuming processing of frame-by-frame editing (See [0003]-[0007])." While Applicant does not necessarily disagree with Examiner's overall characterization of the function of Li, Applicant does respectfully disagree that this function represents or suggests converting a second graphics object into still picture data if an overlap cue indicates overlap between first and second graphics objects.

Li is fundamentally directed at receiving a video signal with a previously applied textual overlay and reversing the overlay process, whereas the present claims are directed at creating a composite OSD for mixture with a video signal. Li operates on a single graphics object, a video frame, to create multiple graphics objects. Thus, Li does not disclose, or have reason to disclose, mixing a stream from an OSD processor with still picture data from a separately processed graphic into a combined overlay for mixture with a received video stream. Therefore, Li, like Terao and Valmiki, fails to disclose "mixing means for mixing said first digital stream, generated by said OSD processor and representing said first graphics object, and said second digital stream, generated from said still picture data converted from said second graphics object from said graphics memory responsive to said overlap cue indicating overlap between the first and second graphics objects, into a video signal," as described in claim 1, and also would fail to address the problem of an OSD processor which is unable to manage two graphic objects that overlap.

In Huang, a "method for controlling various mode of OSD functions is provided in a display system. The method uses a display buffer with plural register for storing frame data, which corresponding to a frame. A memory is used for storing OSD data, corresponding to an OSD window included in the frame. The micro-controller copies the OSD data in the

register which corresponds to the OSD window by an data processing method." (Huang Abstract)

Huang is directed at the problem that conventional OSD circuits lack flexibility in providing multiple windows simultaneously. Huang does not disclose converting one of two overlapping graphics objects into picture data for subsequent mixing with an OSD stream. Therefore, Huang, like Li, Terao, and Valmiki, fails to disclose "mixing means for mixing said first digital stream, generated by said OSD processor and representing said first graphics object, and said second digital stream, generated from said still picture data converted from said second graphics object from said graphics memory responsive to said overlap cue indicating overlap between the first and second graphics objects, into a video signal," as described in claim 1, and also would fail to address the problem of an OSD processor which is unable to manage two graphic objects that overlap.

In view of the above remarks and amendments to the claims, it is respectfully submitted that there is no 35 USC 112 enabling disclosure provided by Valmiki, Terao, Li, or Huang, alone or in combination, that makes the present invention as claimed in claim 1 unpatentable. It is also respectfully submitted that independent claim 10 is allowable for at least the same reasons as claim 1. Since dependent claims 3-7 are dependent from allowable independent claim 1, it is submitted that they too are allowable for at least the same reasons claim 1 is allowable. Thus, it is further respectfully submitted that this rejection has been satisfied and should be withdrawn.

Having fully addressed the Examiner's rejections it is believed that, in view of the preceding amendments and remarks, this application stands in condition for allowance. Accordingly then, reconsideration and allowance are respectfully solicited. If, however, the Examiner is of the opinion that such action cannot be taken, the Examiner is invited to contact the applicant's representative at (609) 734-6804, so that a mutually convenient date and time for a telephonic interview may be scheduled.

No fee is believed due. However, if a fee is due, please charge the additional fee to Deposit Account 07-0832.

Respectfully submitted,
Edouard Ritz

/Brian J. Cromarty/

By: Brian J. Cromarty
Attorney for Applicant
Reg. No. 64018
Phone (609) 734-6804

Patent Operations
Thomson Licensing Inc.
P.O. Box 5312
Princeton, New Jersey 08543-5312

November 30, 2010